SoftwareX 13 (2021) 100619

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

advligorts: The Advanced LIGO real-time digital control and data acquisition system

Rolf Bork [a], Jonathan Hanks [b,a], David Barker [b,a], Joseph Betzwieser [c,a], Jameson Rollins [a,*], Keith Thorne [c,a], Erik von Reis [b,a]

[a] California Institute of Technology, Pasadena, CA 91125, USA
[b] LIGO Hanford Observatory, Richland, WA 99352, USA
[c] LIGO Livingston Observatory, Livingston, LA 70754, USA

ARTICLE INFO

Article history:
Received 31 March 2020
Received in revised form 9 July 2020
Accepted 3 November 2020

Keywords:
Real-time processing
Feedback control
Hardware control
Data acquisition

ABSTRACT

The Advanced LIGO detectors are sophisticated opto-mechanical devices. At the core of their operation is feedback control. The Advanced LIGO project developed a custom digital control and data acquisition system to handle the unique needs of this new breed of astronomical detector. The advligortsis the software component of this system. This highly modular and extensible system has enabled the unprecedented performance of the LIGO instruments, and has been a vital component in the direct detection of gravitational waves.

Code metadata

Current code version	**4.0~pre**
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2020_155
Code Ocean compute capsule	n/a
Legal Code License	GPLv3
Code versioning system	git
Software code languages, required tools and services	C/C++, Perl, Python
Compilation requirements, operating environments	C99, Linux OS
Developer documentation/manual	https://dcc.ligo.org/LIGO-E1200653/public
Support email	jameson.rollins@ligo.org

1. Motivation and significance

The development of long baseline gravitational wave detectors over the last 40 years has been of groundbreaking scientific impact. These sophisticated measurement devices are revolutionizing astronomy and astrophysics by providing an entirely new perception of the universe.

The complexity and scale of the LIGO detectors was novel. While prototype interferometer gravitational wave detectors paved the way, their relative simplicity could not provide a clear roadmap for the ultimate design of the LIGO detectors. This was particularly true for the interferometer feedback control systems at the core of the instruments' operation.

Feedback control enables the LIGO interferometers to maintain the operating point of a highly nonlinear precision optical instrument. Sensors provide information about the state of the interferometers that are fed as error signals to feedback controllers that filter and transform the signals to produce control signals that are fed back to the instrument to actuate on its state. The dynamics of the instruments were extensively modeled before their construction, but the final design of the control loops could not be fully conceived a priori. The feedback controllers needed to be flexible to account for uncertainty in the final feedback control scheme. This motivated a critical decision early in the LIGO design process: to use digital instead of analog feedback control.

* Corresponding author.
E-mail address: jrollins@caltech.edu (J. Rollins).

https://doi.org/10.1016/j.softx.2020.100619

At the time, it was unclear if digital control was feasible for this application. There were concerns that digital control loops would have insufficient bandwidth to control the instrument, or that the analog↔digital conversion processes would inject too much noise into the feedback loops, limiting the sensitivity of the detectors. However, analog electronics are difficult and time consuming to modify, which would have severely limited the rate at which changes to the controllers could have been made and, therefore, the rate of refinement of the instruments. Systems that could have allowed for faster turnaround – by making any potentially desirable parameter changes easily switchable – would have necessarily been complex, difficult to design and maintain, and expensive.

The parameters of digital controllers, on the other hand, can be modified nearly instantaneously, allowing much faster refinement. Further modeling also made clear that excess noise from the digitization and analogization processes could be sufficiently mitigated by proper design of analog signal conditioning, and by expected improvements to the bit depth and noise performance of analog↔digital conversion electronics. The increasing speed of computers would allow the bandwidth and complexity of the controllers to steadily increase over time. Ultimately, it was determined that the ease of quick modification – aided by the promising advancements in performance – made digital feedback the necessary choice.

The digital controllers for the Initial LIGO detectors [1] were handwritten C code that was compiled into dedicated real-time Linux kernels. Modular filter banks that allowed for in situ switching of feedback filters and gains were inserted at key points in the feedback path to allow instrument scientists to change the controller response on the fly. The feedback logic itself was hard-coded and difficult to modify. This was sufficient for Initial LIGO, with its small number of feedback control loops. Advanced LIGO was significantly more complicated and required digital feedback controllers that were more flexible.

For Advanced LIGO [2], a more modular, extensible, and usable system was designed from the ground up. The `advligorts` system gave scientists the ability to visually represent the feedback signal flow and control logic in a more intuitive way using a graphical user interface. The signal flow diagrams could be compiled into real-time code and re-loaded into the front end computers in a matter of minutes, considerably speeding up the turn around time to affect changes on the system.

The `advligorts` system helped bring the new gravitational wave detectors to fruition, enabling the first detection of gravitational waves in 2015 [3].

2. Software description

`advligorts` is the software component of the full Advanced LIGO digital control and data acquisition system (hereafter also referred to as the "real-time system" or "RTS"). The hardware consists of analog-to-digital (ADC) and digital-to-analog (DAC) converters, binary I/O modules, a timing distribution system to clock the ADCs and DACs, and PCIe buses that interface all the hardware to the front end host computers that run the `advligorts` software (see Fig. 1). The `advligorts` software reads from and writes to the hardware, executes all the digital control logic, and passes data to the data acquisition pipeline. Fig. 2 shows a more detailed schematic of this architecture, showing both hardware and software components.

A important feature of `advligorts` is its use of the Experimental Physics and Industrial Control System (`EPICS`), a Free and open-source message passing system [4,5]. `EPICS` provides a standard interface for operator interfaces (LIGO uses the "MEDM" GUI tool) and supervisory control, such as `guardian`, the Advanced LIGO automation platform [6].

2.1. Software architecture

The `advligorts` software consists of three primary components: a patched version of the standard Linux kernel, a *real-time code generator* (RCG), and a suite of data acquisition daemons (DAQD).

The Linux kernel patch, which is simple and small, allows loadable kernel modules to request that the kernel remove a specific CPU from the normal linux process scheduler and hand it over for exclusive use by the module code. The kernel module can then have uninterruptible use of the CPU at full rate.

Typically `advligorts` programs are drawn as signal flow diagrams in MATLAB Simulink [7]. The RCG takes a Simulink model as input, parses it to extract the signal flow graph, and outputs C code that can execute the same signal flow logic. The RCG adds wrapper code for process synchronization and inter-process communication, then compiles the resulting code into an RTS linux kernel module that can be immediately inserted into the running kernel. The RTS module is completely self sufficient at runtime, requiring no services from the kernel or the rest of the operating system. A special RTS module called an *input–output processor* (IOP) handles all I/O processing, reading data from the ADCs and passing it to other RTS modules via a shared memory interface, and writing output data passed from other RTS modules to the DACs. The IOP processes run at 2^{16} Hz, and normal RTS modules can run at any power of two less than that.

Each RTS module employs a set of user space processes to handle user-facing IO tasks. An `EPICS` *input/output controller* ("EPICS IOC" in Fig. 2) process exposes signal monitors and writable parameters of the controller logic to the `EPICS` interface. The "awgtpman" process handles the activation of temporary test point outputs and the processing and insertion of signal injections sent over the network (an interface protocol referred to as "AWG"). Each of these user space processes uses shared memory to exchange data with their respective RTS kernel space processes. Interconnection between RTS processes is achieved either via shared memory for RTS modules in the same kernel, or via a Dolphin PCIe shared memory network [8] for RTS modules on different hosts.

Each front end host computer also runs two processes that collect data from the RTS processes via shared memory ("local_dc") and send the data over the network in 1/16th-second chunks ("data_xmit"). The data acquisition chain receives data from all front ends and collates them into 1/16th-second blocks that are consumed by the DAQD processes which write the data to disk, serve it over the network, or forward it to other streaming processes. In the production environment, the DAQD processes are spread across multiple machines to reduce resource competition.

2.2. Software functionality

`advligorts` leverages MATLAB Simulink to provide a powerful graphical user interface for users to draw signal flow graphs that can be turned into real-time code (see Section 3). Users can drag and drop parts from an extensive parts library and easily connect them together. The parts library includes parts for all supported I/O interfaces, most math operations, matrices, logic gates, switches, oscillators, demodulation, etc., allowing for essentially arbitrary signal processing.

Most of the parameters of all parts are exposed as process variables through the `EPICS` interface. Process variables can take the form of floats, integers, or strings, and are used to control switches, set signal gains and matrix coefficients, etc. The `EPICS` interface also provides monitor test points for the signals passing through the system. Many tools exist to interface with `EPICS`, including python libraries, command line tools, and various graphical programs for creating operator interface screens.

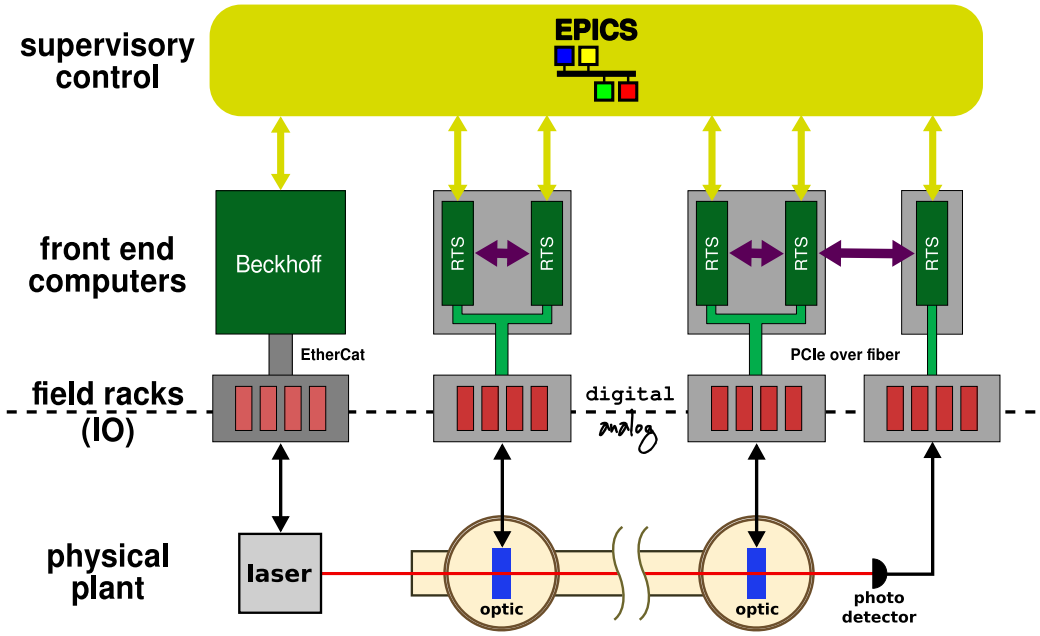


Fig. 1. Overview of Advanced LIGO's digital instrument control system. The physical plant consists of the interferometers themselves, and their sensors and actuators. Signals are digitized and interpolated in the field racks, which are physically separated from the front end computers and connected to them via PCIe fiber. The front end computers handle all real-time control with the `advligorts` software (RTS). Supervisory control and operator interfaces communicate with the RTS via the `EPICS` message passing interface. Beckhoff is a commercial PC-based programmable logic controller used for some slow control tasks. Beckhoff uses the EtherCAT protocol to communicate with hardware devices and `EPICS` to communicate with the rest of the system.

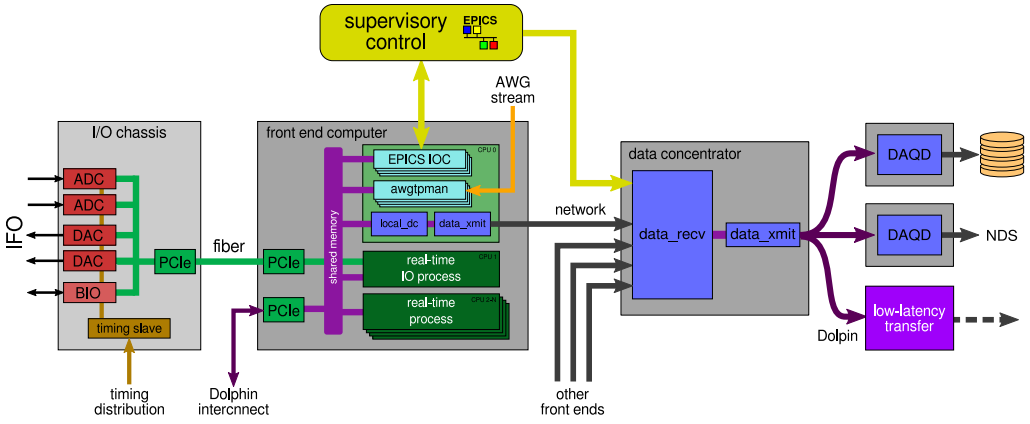


Fig. 2. Architecture of the Advanced LIGO RTS. At left is the I/O chassis that holds ADC, DAC and binary I/O (BIO) cards, and the timing slave that clocks the hardware. The I/O chassis is connected to the front end host computer via a fiber PCIe bus extension. The front end computer runs an RTS version of the Linux kernel, into which multiple RCG-generated RTS kernel modules (dark green) can be loaded. The user space RTS processes are shown in cyan. The data acquisition components are shown in blue. The "data concentrator" collects data from the distributed front end computers and passes the concatenated data to a DAQD process that writes it to disk, a DAQD network data server (NDS), or to the low-latency streaming service. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

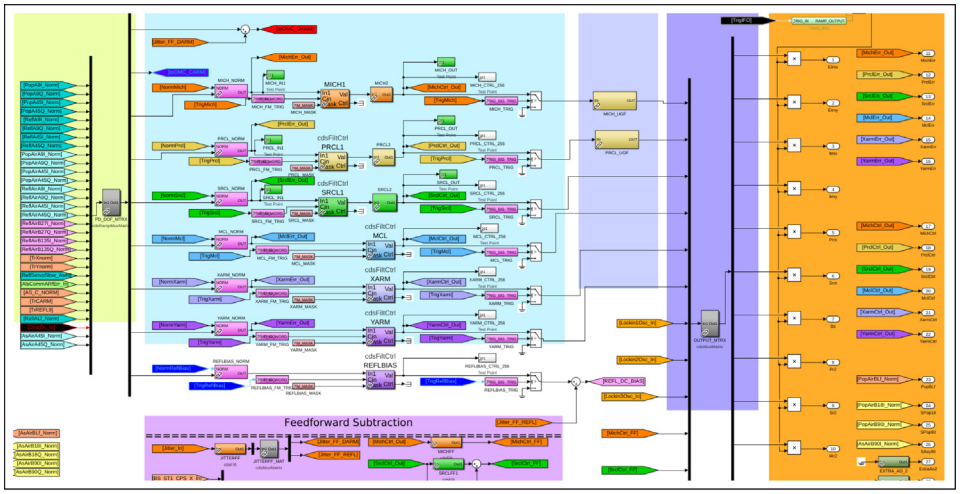
One of the most important parts is the filter module. Filter modules hold a bank of ten addressable digital filters, each with up to 20 poles and zeros, implemented as cascaded second-order-sections. A companion filter design tool called `foton` allows users to design filters from scratch, or draw from a library of common filter functions. Once loaded, individual filters can be engaged or disengaged in situ via a single command from the `EPICS` interface. The filter modules also include excitation inputs, test points, switches for input and output engagement, an additive offset, and a multiplicative gain.

The RCG also allows users to define their own `C` code functions that can be dropped anywhere in the signal flow graph to perform arbitrary signal manipulations.

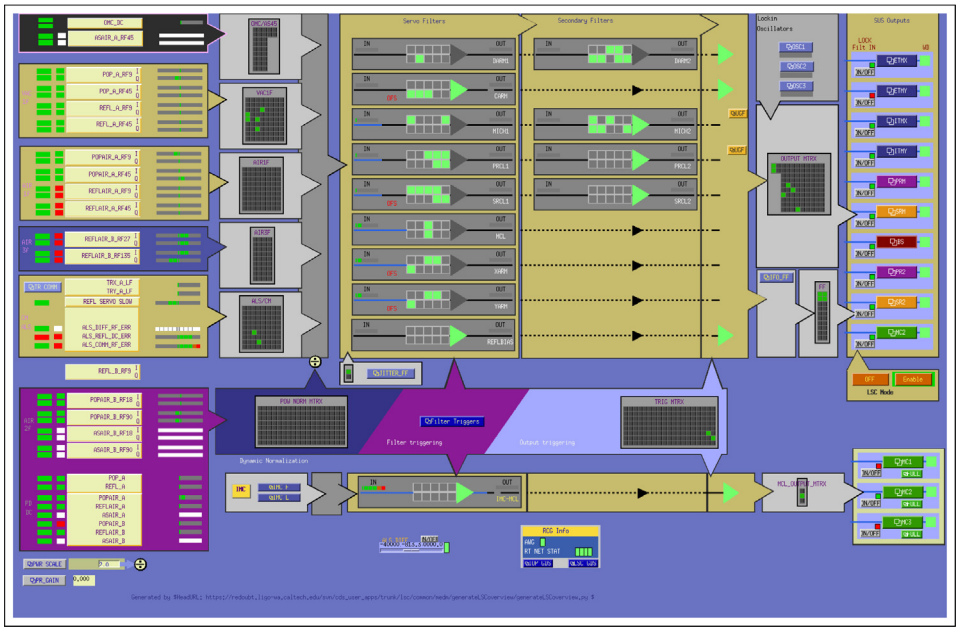
The data acquisition daemon provides a Network Data Service ("NDS") which can be used to access all data acquired by the system in real-time, or to access archival data that has been stored to disk. This service also allows users to request real-time data from virtual test points in the system. The NDS test point interface and the AWG signal injection interface together provide a critical interface for real-time characterization of the system not readily available on other digital signal processing systems. Various supporting applications give scientists the ability to plot the data in both the time and frequency domains, and to make various excition-based measurements of the system.

3. Illustrative examples

Each Advanced LIGO interferometer employs over 120 RTS models, spread across nearly three dozen front end computers. The data acquisition systems process roughly 10k "fast" channels



(a) Excerpt of Simulink code for the LSC real-time controller



(b) Screenshot of the operator interface for LSC real-time controller

Fig. 3. A small excerpt of Simulink code for the length sensing and control (LSC) subsystem, and a screen shot of the corresponding operator interface. See text for description of the components and structure of the code.

(with sample rates from 512 - 16k Hz) and nearly 300k "slow" channels (16 Hz, EPICS process variables). The front end computers are standard off-the-shelf Intel Xeon x86_64 servers, with each RTS process given at least 1 GB of RAM and a single, physical core running at roughly 3 GHz.

Fig. 3a shows an excerpt of the Simulink code used to program the real-time controller for the LIGO length sensing and control subsystem which controls the lengths of all the various optical cavities in the Advanced LIGO interferometers. This model runs at 16k Hz, taking as input the RF-demodulated photodetector error signals. Its outputs are fed over the Dolphin network to separate RTS models that control the suspended optics.

Signal flow is drawn from left to right, with outputs for this block at the far right. The diagram is less busy that it might otherwise be because of the liberal use of GOTO/FROM connections that eliminate the need for drawn lines connecting the output of one block to the input of another. Filter modules are labeled "`cdsFiltCtrl`". At the left of the diagram, a gray box represents a matrix (labeled "PD_DOF_MTRX") that handles "rotation" of sensor input signals into the canonical longitudinal degree of freedom basis for the interferometers. Near the right another matrix ("OUTPUT_MTRX") handles rotation of signals in the degree of freedom basis into the basis of output suspension drive actuators.

4. Impact

Construction of the Advanced LIGO detectors was completed in 2014. The first gravitational wave was detected in September 2015 [3]. The flexibility, modularity, and ease of programming allowed by the `advligorts` system was critical for the rapid commissioning of the detectors down to unprecedented sensitivities. The ongoing success of LIGO (there have been dozens more confirmed detections since GW150914 [9]) is a testament to the robustness of this system.

In August 2017, LIGO made the first ever detection of gravitational waves from a binary neutron star merger [10]. With help from the Virgo detector [11], the merger was localized to an area

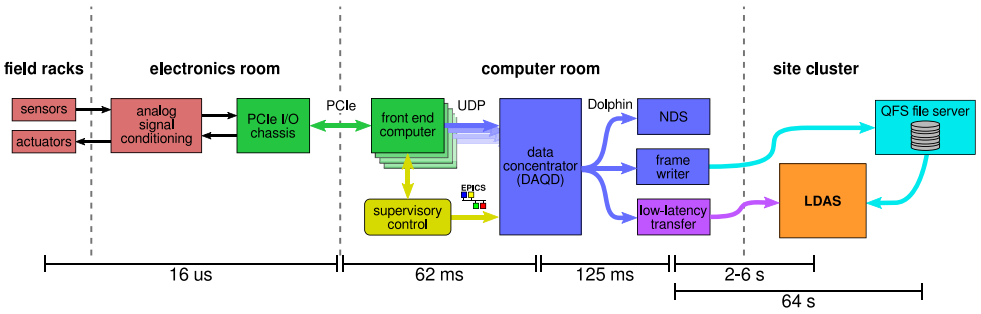


Fig. 4. Advanced LIGO's data processing pipeline. Times shown at the bottom are sample latencies. PCIe, UDP, and Dolphin refer to the transport fabrics used. The gravitational wave searches are conducted on the LIGO "LDAS" computer clusters, which receive data from the observatories within seconds.

in the sky of roughly 30 deg^2 [12]. Within hours of detection and localization, alerts had been sent to the astronomical community and dozens of electromagnetic observatories across the world, who then made the first observation of a gamma-ray burst emitting kilonova [12]. These observations were made possible by a data acquisition system that was able to reliably deliver strain data from the detectors to the search pipelines within a matter of seconds (see Fig. 4). The `advligorts` data acquisition system is the source of this entire pipeline.

Advanced LIGO has already undergone one significant upgrade, installing a squeezed light source which reduces quantum noise at high frequencies [13]. And the more significant "A+" upgrade will commence at the end of the current observing run, adding another new subsystem to shape the quantum noise suppression from the squeezed light source, and further increasing the sensitivity of the current detectors to less than 10^{-22} m/$\sqrt{\text{Hz}}$. These upgrades are enabled by the flexibility and extensibility of `advligorts` framework.

With the success of LIGO, use of the `advligorts` system is spreading throughout the gravitational wave community. The Japanese project KAGRA [14] has adopted the full Advanced LIGO digital control and data acquisition system for control of their underground gravitational wave detector. `advligorts` is also used in the GEO 600 project, the Caltech 40 m prototype, the MIT LASTI prototype, the AEI 10 m prototype, as well as in dozens of smaller laboratories around the world to control a variety of table-top opto-mechanical experiments.

5. Conclusions and future development

LIGO is expected to operate gravitational wave observatories for at least the next two decades. In that time, a third identical detector is expected to be completed in India [15]. Beyond this, a new generation of ground-based detectors is being planned to extend our reach out to the edge of the observable universe [16–18]. These new detectors will have even more complex controls challenges, and the `advligorts` system will need to continue to evolved and adapt to meet their needs.

Development is ongoing to improve the usability and extend the functionality of `advligorts`. A key development track aims to see if the need for a specially-patched linux kernel can be eliminated, by leveraging new features in the standard linux kernel that allow for CPU-isolation for privileged processes. Developers have also been experimenting with running RTS processes in user space. Unprivileged user space execution would not be for real-time operation, but could allow for running code faster than real-time for simulation or testing purposes.

A project is also underway to explore the possibility of executing neural networks within RTS processes. Neural networks trained on simulations and archived data could be ported into RTS processes to enable machine learning experimental control. There is potential for training and updating these networks in real-time, for more sophisticated reinforcement learning applications.

The data acquisition pipeline is being improved to increase the throughput and accessibility of larger amounts of data. Currently, only a small subset of the full channel data is available in low latency, limiting the amount of data characterization that can be done for the low-latency search pipelines. It should be possible to dump the full data pipe into the local LDAS clusters at a fraction of the current time, decreasing latencies for multi-messenger searches and potentially leading to more ground-breaking discoveries.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

LIGO was constructed by the California Institute of Technology and Massachusetts Institute of Technology with funding from the National Science Foundation, United States and operates under Grant No. PHY-0757058. Advanced LIGO was built under award PHY-0823459.

References

[1] Abbott BP, et al. LIGO: the laser interferometer gravitational-wave observatory. Rep Progr Phys 2009;72(7):076901. http://dx.doi.org/10.1088/0034-4885/72/7/076901, 0711.3041.
[2] Aasi J, et al. Advanced LIGO. Classical Quantum Gravity 2015;32(7):074001. http://dx.doi.org/10.1088/0264-9381/32/7/074001.
[3] Abbott B, et al. Observation of gravitational waves from a binary black hole merger. Phys Rev Lett 2016;116(6). http://dx.doi.org/10.1103/physrevlett.116.061102.
[4] Experimental Physics and Industrial Control System (EPICS). http://www.aps.anl.gov/epics/.
[5] Dalesio LR, Kraimer MR, Kozubal AJ. EPICS architecture. In: 4th international conference on accelerator and large experimental control systems. Joint accelerator conferences website (JACoW). 1991.
[6] Rollins JG. Distributed state machine supervision for long-baseline gravitational-wave detectors. Rev Sci Instrum 2016;87(9):094502. http://dx.doi.org/10.1063/1.4961665.
[7] MATLAB Simulink. https://www.mathworks.com/products/simulink.html.
[8] Dolphin Interconnect Solutions. https://dolphinics.com/.
[9] Abbott B, et al. GWTC-1: A gravitational-wave transient catalog of compact binary mergers observed by LIGO and virgo during the first and second observing runs. Phys Rev X 2019;9(3). http://dx.doi.org/10.1103/physrevx.9.031040.
[10] Abbott B, et al. GW170817: Observation of gravitational waves from a binary neutron star inspiral. Phys Rev Lett 2017;119(16). http://dx.doi.org/10.1103/physrevlett.119.161101.
[11] Acernese F, et al. Advanced virgo: a second-generation interferometric gravitational wave detector. Classical Quantum Gravity 2014;32(2):024001. http://dx.doi.org/10.1088/0264-9381/32/2/024001.

[12] Abbott BP, et al. Multi-messenger observations of a binary neutron star merger. Astrophys J 2017;848(2):12. http://dx.doi.org/10.3847/2041-8213/aa91c9.
[13] Tse M, et al. Quantum-enhanced advanced LIGO detectors in the era of gravitational-wave astronomy. Phys Rev Lett 2019;123(23). http://dx.doi.org/10.1103/physrevlett.123.231107.
[14] Aso Y, Michimura Y, Somiya K, Ando M, Miyakawa O, Sekiguchi T, Tatsumi D, Yamamoto H. Interferometer design of the KAGRA gravitational wave detector. Phys Rev D 2013;88(4). http://dx.doi.org/10.1103/physrevd.88.043007.
[15] Abbott BP, et al. Prospects for observing and localizing gravitational-wave transients with advanced LIGO, advanced virgo and KAGRA. Living Rev Relativ 2018;21(1). http://dx.doi.org/10.1007/s41114-018-0012-9.
[16] Adhikari RX, et al. A cryogenic silicon interferometer for gravitational-wave detection. 2020, 2001.11173. URL: http://arxiv.org/abs/2001.11173.
[17] Abbott BP, et al. Exploring the sensitivity of next generation gravitational wave detectors. Classical Quantum Gravity 2017;34(4):044001. http://dx.doi.org/10.1088/1361-6382/aa51f4.
[18] Punturo M, et al. The einstein telescope: a third-generation gravitational wave observatory. Classical Quantum Gravity 2010;27(19):194002. http://dx.doi.org/10.1088/0264-9381/27/19/194002.